An Interactive, Constraint-Based System for Task Allocation in an Academic Environment

Ryan Lim, Praveen Guddeti, and Berthe Y. Choueiry
Constraint Systems Laboratory
Computer Science & Engineering
University of Nebraska-Lincoln
{rlim, vgudetti, choueiry}@cse.unl.edu

1 INTRODUCTION

This paper describes an interactive system for the management of Graduate Teaching Assistants (GTAs) in our department. The system is based on Constraint Processing techniques and is operated through web-based interfaces. The task is to assign GTAs, based on their qualifications, availability, and preferences, to academic tasks over a semester such as grading, supervising labs and recitations, and teaching introductory classes. Typically, every semester, the department has about 70 different academic tasks and can hire between 25 and 40 GTAs. The problem is often tight and sometimes over-constrained. In the past, this task has been performed manually by members of the staff and faculty. Tentative schedules were iteratively refined based on feedback from faculty and the GTAs themselves, in a tedious and error-prone process lingering over 3 weeks. It was quite common to have in the final hand-made assignment a number of conflicts and inconsistencies, which negatively affected the quality of our program. For instance, when a course is assigned to a GTA with little knowledge of the subject matter, the course's instructor has to take over a large portion of the GTA's job and the GTA has to invest considerable effort to adjust to the situation. Moreover, students in the course may receive diminished feedback and unfair grading.

We have built web-based interfaces to streamline the collection of data and specification of constraints. Further, we have implemented a number of constraint-based functionalities that assist the human manager in generating solutions interactively and automatically (Lim, Guddeti, & Choueiry 2004). The modeling efforts started in Spring 2001. Various versions of the prototype have been used since Fall 2001. This system has effectively reduced the number of obvious conflicts with positive effects on the quality of our academic program. It has also decreased the amount of time and effort spent on making the assignment and gained everyone's approval.

2 SYSTEM ARCHITECTURE

The current system has the following main components (Fig. 1): web-interfaces for data acquisition, a relational database for storing the collected data, a number of search algorithms for problem solving, and the ability to generate reports. The implementation is heterogeneous: the database

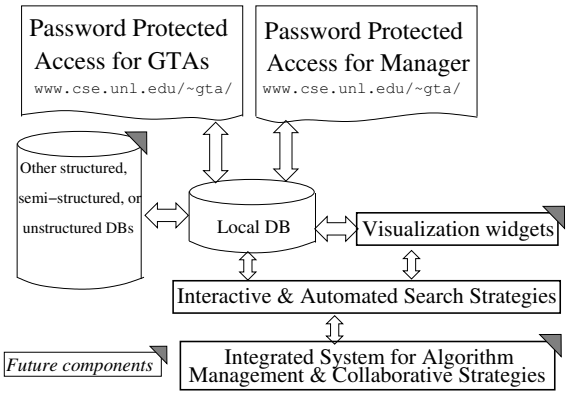

Figure 1: *System Architecture.*

is a MySQL database, the web-interfaces are implemented in PHP, and the search algorithms are in Common Lisp and C++. The web interfaces allow the applicants and the manager to specify the information used for determining the consistency of a solution. This includes the type and load of courses, the qualifications of GTAs, their hiring capacity, and their preferences for each course on a scale from 5 to 0 ('5 Best choice,' '4 Favorite,' '3 Qualified,' '2 Able to handle,' '1 Avoid if possible,' '0 Cannot handle'). This information, in addition to the class schedule (retrieved from a departmental database), is stored in the MySQL database. While the database server and the web-interfaces are not yet intelligent components, they are essential for collecting the necessary data and for monitoring and using the implemented mechanisms. They are also necessary for the future developments we foresee. These developments include: (1) constraint-based, secure interfacing with university-wide (more or less) structured databases, (2) data-visualization widgets for interaction with users, and (3) an integrated system for algorithm management and collaboration strategies among the algorithms.

In addition to allowing the manager to specify the constraints and browse the students' data, the manager's web-interface provides two operational modes for problem solving: interactive and automatic. In both modes, the problem is modeled as a constrained optimization problem. We

describe first the modeling, then the problem solving processes.

3 MODELING

We choose to express the courses as the variables of a Constraint Satisfaction Problem (CSP) and the GTAs as the values that these courses may take. We express the constraints that restrict the acceptable assignments as unary, binary, and non-binary constraints. For example, unary constraints specify students' availability and qualifications; binary constraints specify courses that overlap in time or must share the same GTA; and non-binary constraints specify capacity limits (i.e., the total load assigned to a given GTA cannot exceed his/her hiring capacity). In practice, the problem is almost always tight and often over-constrained, but this is not known a priori. The goal is to cover as many classes as possible while maximizing the preferences of the GTAs. Because the hard constraints cannot be violated, the problem cannot be modeled as a MAX-CSP (Freuder & Wallace 1992). Thus, the goal is to find the longest partial and consistent solution (as a primary optimization criterion) that maximizes GTA preferences (as a secondary optimization criterion). In (Glaubius 2001; Glaubius & Choueiry 2002c), we describe the CSP model and some criteria for maximizing GTA preferences. In (Glaubius & Choueiry 2002a; 2002b), we reformulate some of the non-binary constraints into binary ones and demonstrate the benefits of this.

4 INTERACTIVE DECISION MAKING

In the interactive mode, the manager can examine the problem from two perspectives and switch between them at any point: a list of courses with a pull-down menu of selectable GTAs, and vice-versa. Fig. 2 shows how the list of GTAs suitable for a given course is displayed to the user. Here, the

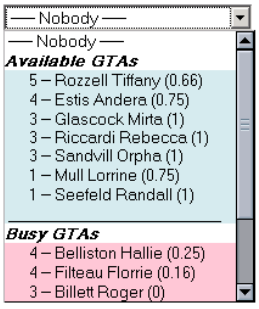

Figure 2: *Inspecting available and busy GTAs for a given course.*

upper portion displays the GTAs that can be assigned to the course without reservation. In terms of the CSP, these are the values in the current domain of the variable. The lower portion lists the GTAs who are appropriate but are 'busy' in other assignments. These are the values eliminated from the domain of the variable by constraint propagation. In each portion, the GTAs are listed in decreasing preference order, as a primary sorting criterion, then in increasing lexicographical of the GTA's last name, as a secondary criterion. Next to the name, the 'current' capacity of each GTA is displayed (which is the hired capacity of the GTA discounted by the load of his/her other assignments). Every time a new assignment is made, a full arc-consistency is executed over the unassigned variables to filter their domains. The manager can easily undo or change assignments. The domains of all the variables are then recomputed from scratch while maintaining the intermediate selections. We decided against using a dynamic arc-consistency algorithm (Bessière 1991) to avoid the use of additional data structures and because the solution adopted is quite efficient in practice. Naturally, this decision may need to be reconsidered if the application size and characteristics change.

5 AUTOMATED SEARCH ALGORITHMS

We have developed and tested various search strategies. The use of real data to compare the behavior and performance of the search strategies has allowed us to identify new characterizations of these search procedures as documented in (Zou & Choueiry 2003b; 2003a; Zou 2003). The search strategies currently available are a heuristic backtrack search (BT), a stochastic local search (with random-walk to escape from local optima and random restarts to recover from them), a multi-agent search as an 'extremely' decentralized local search, and a randomized backtrack search with dynamic restart (Guddeti & Choueiry 2004). Interestingly, although BT is theoretically sound and complete, the size of the search space makes such guarantees meaningless in practice. Indeed, in a problem with 60 variables and approximately 30 values per variable, the shallowest level that the backtracking reached after 24 hours (depth level 51) was not significantly better than the level it reached after 1 minute (depth level 55). Further, we found that the multi-agent search was consistently the only technique capable of solving tight but solvable problems, thus demonstrating an unequaled immunity to local optima. We also identified a deadlock phenomenon that dramatically undermines its stability in over-constrained problems.

6 CONCLUSIONS and FUTURE WORK

GTA assignment is a particularly difficult task for humans because of the wealth of hard constraints that must be satisfied. On the other hand, the manager typically has, at his/her fingertips, many subjective information about the part GTAs, the courses, and the instructors that cannot be formally expressed. In this application, *computers proved to be most effective in exactly the same tasks at which humans fail* (i.e., keeping track of hard constraints), *and vice versa* (i.e., quickly evaluating alternatives and making compromises). Since we started using the system in Fall 2001, the assignments generated by the system were used as first solutions finely tweaked by the manager. Invariably, the final decisions were made quickly, and the assignments proved more stable and more satisfactory to everyone in the department.

In the future we will design hybrid search procedures that work seamlessly together, and design visualization widgets that enable the manager to actively guide and monitor the progress of the search process and mechanisms to compose partial solutions built manually or with search.

Acknowledgments

The development of the algorithms is supported by grant #EPS-0091900 of the National Science Foundation. The development of the database and web interfaces is supported by the Department of Computer Science & Engineering of University of Nebraska-Lincoln. Several staff members of our department provided domain expertise and practical feedback, and several students helped building the system.

References

Bessière, C. 1991. Arc-Consistency in Dynamic Constraint Satisfaction Problems. In *Proc. of AAAI-91*, 221–226.

Freuder, E. C., and Wallace, R. J. 1992. Partial Constraint Satisfaction. *Artificial Intelligence* 58:21–70.

Glaubius, R., and Choueiry, B. Y. 2002a. Constraint Modeling and Reformulation in the Context of Academic Task Assignment. In *Working Notes of the Workshop on Modelling and Solving Problems with Constraints, ECAI 2002*.

Glaubius, R., and Choueiry, B. Y. 2002b. Constraint Modeling and Reformulation in the Context of Academic Task Assignment . Poster presentation at the Fifth International Symposium on Abstraction, Reformulation and Approximation (SARA 2002).

Glaubius, R., and Choueiry, B. Y. 2002c. Constraint Modeling in the Context of Academic Task Assignment. In Hentenryck, P. V., ed., *Proceedings of* 8^{th} *International Conference on Principle and Practice of Constraint Programming (CP'02)*, volume 2470 of *Lecture Notes in Computer Science*, 789. Ithaca, NY: Springer Verlag.

Glaubius, R. 2001. A Constraint Processing Approach to Assigning Graduate Teaching Assistants to Courses. Undergraduate Honors Thesis. Department of Computer Science and Engineering, University of Nebraska-Lincoln.

Guddeti, V. P., and Choueiry, B. Y. 2004. An Improved Restart Strategy for Randomized Backtrack Search. In *Under review*.

Lim, R.; Guddeti, V. P.; and Choueiry, B. Y. 2004. An Interactive System for Hiring and Managing Graduate Teaching Assistants. In *Conference on Prestigious Applications of Intelligent Systems (PAIS 04)*, 5 pages.

Zou, H., and Choueiry, B. Y. 2003a. Characterizing the Behavior of a Multi-Agent Search by Using it to Solve a Tight, Real-World Resource Allocation Problem. In *Workshop on Applications of Constraint Programming*, 81–101.

Zou, H., and Choueiry, B. Y. 2003b. Multi-agent Based Search versus Local Search and Backtrack Search for Solving Tight CSPs: A Practical Case Study. In *Working Notes of the Workshop on Stochastic Search Algorithms (IJCAI 03)*, 17–24.

Zou, H. 2003. Iterative Improvement Techniques for Solving Tight Constraint Satisfaction Problems. Master's thesis, Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE.